8 Loading and Transforming Documents

It is now time to turn our attention to the workhorse issues of document warehousing: loading and transforming documents. At this stage, documents are preprocessed, if necessary, to ensure that they are in a character format and language appropriate for the tools that will later perform text analysis. Documents are then indexed for both full text and themes. Depending upon the needs of document warehouse users, documents may also need to be classified, grouped with similar documents, and summarized. This chapter will examine each of these steps in the following order:

- Internationalization and character set issues
- Translating documents
- Indexing texts
- Classifying documents
- Clustering documents
- Summarizing text

We will examine language differences in the first two topics, with an emphasis on the importance of the Unicode standards and on the uses—and limits—of machine translation. Since the basics of full text and thematic indexing have already been discussed in Chapter 4, we will now look into customizing indexing with specialized thesauri and stoplists. Classification of documents can be done in

Figure 8.1 An overview of document loading and transformation.

two ways, using single labels or using multidimensional taxonomies. Similarly, the problem of grouping documents into clusters can be approached in at least three distinct ways, and we will examine those and discuss their appropriate uses as well. Finally, we will look in detail at different approaches to text summarization. Figure 8.1 shows the basic steps in the loading and transformation process. First, preprocessing steps are performed to ensure documents are in a form suitable for text analysis. Then full text and thematic indexing is done followed by higher level text analysis operation, such as classification, clustering, and summarization.

Internationalization and Character Set Issues

Rapid globalization of economies is bringing new issues right into the middle of many businesses operations. The European Community has had to deal with it since its inception, and has found the cost of publishing every official document in 11 different languages simply astounding. What does this mean to document warehouse practitioners? Mainly that they will now need to meet several different types of multilingual needs, including:

- Business intelligence sources in multiple languages
- Translations of internal documents, such as procedure manuals and user guides
- Government publications, including regulations and advisory notices
- Contracts and other interbusiness documents

Business intelligence sources in different languages are essential for a business with operations in multiple countries. If we depend upon domestic sources for information about a foreign business climate, then we run the risk of missing details not covered by the domestic press. There is also the problem of relevant cultural differences that may be uncovered from resident sources that are lost in local coverage.

Multinational companies have always had to deal with multiple translations of documents. The document warehouse does not create additional problems, it just brings some issues to the forefront. For example, if all documents within the warehouse should be equally accessible (assuming appropriate security controls), then the warehouse will need to store, and client applications will need to render, multiple alphabets.

Managing operations across borders also requires the ability to track and abide by the laws of each host country. While international agreements—such as the European Union regulations and the North American Free Trade Agreement—have eased international business, local regulations must also be tracked. Some nations, such as Germany, have more extensive laws governing commerce than other countries. In these cases, not only do the local divisions of a company need to understand these laws, but other divisions that work with them should know them as well. For example, if a manufacturer's facility in Ireland is planning on increasing production, it could affect distribution centers in Germany. Will the German center need to increase staff or make capital investments because of limits on hours of business operations? While the details of such questions will probably need to be addressed by the local division, understanding the operating environment of business units in other nations could prove to be an advantage.

Of course, government regulations are not the only source of international documents. Business-to-business dealings will generate plenty of contracts, agreements, and other documents that should be managed within the document warehouse.

Meeting these and other needs will require that document warehouse designers and developers deal with two language-related issues: character sets and machine translation.

Coded Character Sets

Coded character sets are used to represent alphabets in computers. Since digital computers fundamentally represent all information using binary numbers, characters need to be mapped into a numeric representation. The two most commonly used are ASCII and Unicode. The former is the older of the two and has long been used in countries with the Latin alphabet. Unicode was developed in response to the need to represent other types of alphabets. Syllabaries use a single character to represent a syllable, such as the kana system used in Japanese to supplement the Chinese characters used in the language. Another alphabet form is ideographic. These systems, such as Chinese, use a single text element to represent an entire word. As Figure 8.1 depicts, modern-day writing systems have evolved and branched off from a variety of earlier systems. The design goal of Unicode was to represent any text element from any language; consequently it is a much richer character set than ASCII.

ASCII Characters

The American Standard Code for Information Interchange (ASCII) was originally a 7-bit character set able to represent 128 letters, numbers, and symbols. The current de facto standard is 8-bit ASCII, which can represent 256 characters. The high-byte characters (from 128 to 255) do not have a standardized character mapping and have been used for formatting features, such as italics,

graphics, and non-Latin characters. Unfortunately, 256 possible characters are not enough to represent all the needed symbols, so Unicode was designed to address this shortcoming.

Unicode

The Unicode coded character set was developed with three design goals in mind:

- Universality
- Uniqueness
- Uniformity

The code was designed as a universal representation system for all written languages, modern and ancient. Each text element has one and only one encoding in Unicode. Also, all characters are represented uniformly in a fixed width representation. The default 16-bit encoding provides for the representation of more than 65,000 characters. Almost 50,000 characters, symbols, and ideographs have been assigned Unicode codes. An extension to Unicode, UTF-16, is a mechanism for providing representations for up to one million more characters. Unicode also uses a single encoding for characters shared across languages, such as those in Chinese, Japanese, and Korean.

It should be noted that Unicode represents abstract characters and does not specify how those characters should be rendered on paper or an electronic display. A glyph, or rendering of a text element, is outside the scope of Unicode.

Most new computing standards, such as XML, and major computing vendors have adopted Unicode. Since the Unicode character set uses the same numbers to encode the Latin alphabet as ASCII, conversion between the two is relatively straightforward. Document warehouse developers will primarily need to concern themselves with ensuring that client browsers support the character sets needed to display the languages found in the warehouse.

Translating Documents

If documents may be added to the warehouse from multiple languages, then designers will need to address translation and language tracking issues. There are basically three issues that need to be addressed:

- Language identification
- Language translation
- Document storage options

Language identification is the first step in managing language issues. There are several options in language translation, and—as usual—your choice depends upon your particular needs. Finally, since translations yield new documents, warehouse designers will need to specify how these new documents are treated within the warehouse.

Language Identification

The three main ways in which language is identified in document warehousing and other text mining operations are:

- Language identification programs
- Search engine restrictions
- Document metadata

Each method has its advantages, and all three can be used reliably in the document warehouse.

We humans can quickly identify text written in our own language, even if we do not understand the content. Take the following example from *Gray's Anatomy*:

> The part of the choroid plexus seen in the descending cornu is formed in exactly the same way, viz, by an ingrowth of the vessels of the pia mater into the cavity, pushing the ependyma before it, at a part of the wall of the horn where there is a similar absence of nervous tissue where it consists simply of pia mater and ependyma in close contact. (Henry Gray. 1977. *The Classic Collector's Edition Gray's Anatomy*, New York: Bounty Books)

Although many of the terms are foreign to most of us, there are enough linguistic clues to know that this is English. First, the Latin alphabet is used. Second, common English words such as *the*, *in*, *at*, *of*, *there*, and *where* appear throughout the passage. Finally, there are morphological clues. The word *formed* ends in *-ed*, making it likely a past tense verb. It is closely followed by a word ending in *-ly* making that word a likely adverb and increasing the likelihood that the word ending in *-ed* is in fact a verb. Just as humans can identify a language without understanding the text, so can text analysis programs.

Language identification programs are generally trained with a sample set of documents in a particular language. Using frequently occurring words and character sequences, these programs can develop profiles of languages and reliably identify a document's source language. The language identification tool in the IBM Intelligent Miner for Text suite, is preconfigured to identify 14 languages:

Brazilian

Catalan

Danish

Dutch

English

Finnish

French

German

Icelandic

Italian

Norwegian

Portuguese

Spanish

Swedish

The statistical techniques that are used with language identification tools generally allow for users to develop identification profiles for other languages. This process usually entails creating training sets of documents in the target language and running the language identification program in a training mode.

The second method for identifying languages is to take advantage of search engine options to restrict searching to a specified language. All documents that are returned from those searches are guaranteed to be in the selected language. Of course, this technique does not help when dealing with internal documents but a similar principal applies. Extraction programs that collect documents may be written to target servers where a single language predominates. For example, a multinational firm can use different processes to collect documents from their London, Rome, and Amsterdam sites so that documents in different languages are kept partitioned before being loaded into the warehouse.

The third method is to use document metadata. The Dublin Core metadata standard includes a language specification for the specified document. For example, a fictional introduction to text mining might include the following metadata specified by the Dublin Core and implemented using the Resource Description Format (RDF)

```
<?xml version="1.0"?>
  <rdf:RDF
       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         <dc:title> Introduction to Text Mining</dc:title>
       <dc:creator> Mary Jones </dc:creator>
       <dc:creator> Bob Smith </dc:creator>
        <dc:subject>
              Text Mining;
```

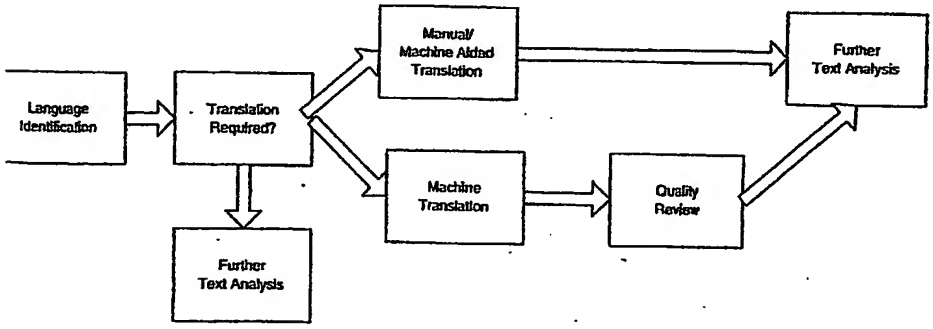
```
            Clustering;
            Summarization;
            Feature Extraction
        </dc:subject>
        <dc:publisher> Association of Text Miners </dc:publisher>
        <dc:date> 2000-08-15 </dc:date>
        <dc:format> text/html </dc:format>
        <dc:language> en </dc:language>
      </rdf:Description>
    </rdf:RDF>
```

The XML entity `<dc:language> en </dc:language>` identifies English as the document's language. Now, it should be noted that the metadata could be specified in other than the language of the document. For example, the tag `<rdf:li xml:lang= >` identifies the language of the metadata, allowing document creators to describe the contents of the document in multiple languages, thus aiding searchers using those other languages.

Language Translation

If language translation is supported in a document warehouse, additional processing and flow of control support is required. As Figure 8.2, shows, once documents have reached a staging area the following steps must be performed:

- Identify the language of the document.
- If the language is to be translated, determine if manual or machine translation will be used.
- If manual translation is selected, add the document to the manual translation queue for the document's language.
- If machine translation is selected, execute the translation program.



gure 8.2 Translation adds several steps to processing the document stream.

- If the full document is to be stored, add it to the document stream for further processing.
- If only a summary of the translated document will be stored, execute a summarization program and add the summary to the document stream for further processing.

Of all these steps, the choice of manual versus machine translation is perhaps the most important. Manual translation offers the best quality translation but is slower and significantly more costly than automatic translation. Machine translation is generally faster but, in general, the reader will only get the gist of the document without a thoroughly accurate translation.

An alternate methodology to the one described above is to store documents in their native language, translate queries, and then provide summaries in the language of the query. If the document is of sufficient interest to the reader, then it can be completely translated. This approach is most appropriate when automated translation is of insufficient quality, and the cost of translating a large volume of documents is prohibitive.

Manual Translation

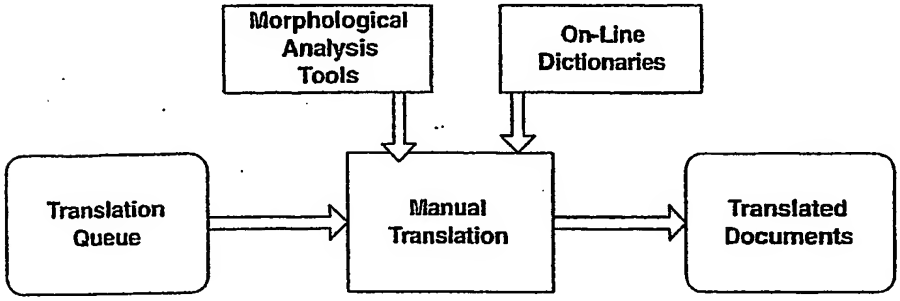
Manual translation is sometimes the best option for ensuring high-quality information in the document warehouse. Machine-aided translation (MAT) provides some automated support for humans through the use of online dictionaries, morphological analysis, and other text processing tools. In the case of MAT, human translators can increase productivity while still controlling quality.

Another option with regard to manual translation is to let a translation program make a first pass at the translation, and then have the human translator finalize the translation. Again, the final quality assurance measures rest with the human translator. As Figure 8.3 shows, there are different options for configuring a manual translation environment.

Machine Translation

The early and persistently elusive goal of machine translation is fully automatic high-quality translation (FAHQT). The ideal translation system works independently of humans yet produces translations at least as good as a human translator. To accomplish this task, the translation system must deal with ambiguity, polysemy, and idioms, among other challenges. Needless to say, we have not yet achieved FAHQT. What has been discovered, however, is that there is a definite tradeoff between the complexity of the translation system and the quality of the translation. Three general approaches, in increasing level of complexity, are:

(a) Manual Translation



(b) Machine Aided Translation

Figure 8.3 Manual translation can be done independently (a) or with the use of machine-aided translation tools (b).

- Direct translation
- Transfer approach
- Interlingua approach

Direct translation was the earliest, and continues to be the most common, design strategy. The transfer approach and interlingua approach were both developed to overcome limitations of earlier approaches.

Direct translation uses a word-for-word approach to translation. As Figure 8.4 shows, a text in a source language is mapped to a target language, using a bilingual dictionary and basic rules for reordering phrases.

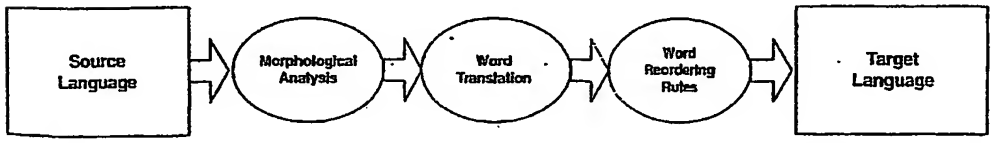


Figure 8.4 Direct translation is the most basic translation technique, using only a dictionary and some rewrite rules.

Table 8.1 English-to-Spanish Substitution Rules

ENGLISH PHRASE	CORRESPONDING SPANISH PHRASE
Adjective Noun	Noun Adjective
Noun1 Noun2	Noun2 "de" Noun1
Adjective1 Adjective2 Noun	Noun Adjective2 Adjective1
Adjective Noun1 Noun2	Noun2 Adjective "de" Noun1

Although fast and efficient, direct translation does not perform any analysis of content or try to resolve ambiguities. It has been successfully used with languages that have similar grammatical structures, for example, English and Spanish. One study of a commercial machine translation program (Gimenez and Forcada 1998) found the substitution rules in Table 8.1 were used in an English-to-Spanish translation program.

The transfer approach improves upon the basic dictionary lookup philosophy of the direct approach by adding syntactic analysis. As Figure 8.5 shows, the second step of this method—the transfer—uses syntactic rules to determine the sentence structure of the target sentence. In the final phase, morphological rules are applied to create the final word forms in the target language, and grammar rules are applied to determine the appropriate phrase and sentence structures. Like the direct approach, the transfer method works only on a single sentence at a time and does not perform semantic analysis. The technique with the most emphasis on semantics is the interlingua approach.

The interlingua approach uses a special language-neutral stage, called the interlingua. The purpose is of the interlingua is to act as a universal semantic representation scheme. Rather than mapping words from the source language onto words in the target language and then rearranging word order according to the grammar of the target language, the interlingua represents the meaning of the source text and then synthesizes text in the target language. Figure 8.6 shows the basic structure of the interlingua approach.

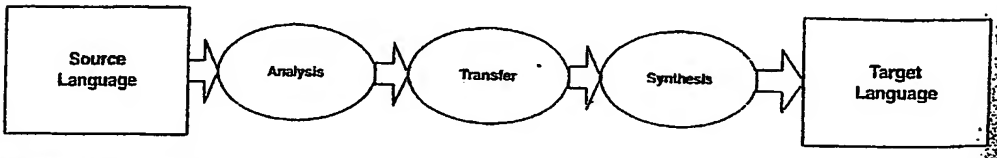


Figure 8.5 The transfer approach includes syntactic analysis and source-to-target language transfer rules.

Although theoretically appealing, this method has not yielded significant results (Hausser 1998). The most significant problem has been finding a suitable interlingua. Proposals include a logic-based language, an artificial language such as Esperanto, and a set of semantic primitives. The use of both logic-based languages and semantic primitives have been extensively studied in Artificial Intelligence (AI), but a comprehensive representation scheme built on these approaches has yet to be developed.

Partial Translations of Structured Texts

In addition to translating entire texts, semistructured texts lend themselves to partial translations. For example, in a financial report it may be sufficient to translate only column and row headings used in tables of financial data. For longer documents, abstracts or executive summaries could be translated by machine to provide the main gist of the document to the reader, while leaving the rest of text to be translated only if there is a specific need.

Limits of Machine Translation

Whether you choose full or partial translation, there are limits to the quality of the translation that end users should be aware of.

First, not all terms used in a source document will have entries in the bilingual dictionary. This is especially true for scientific and technical terms. Many software packages do, however, allow users to add additional terms for specialized vocabularies.

Second, outside of restricted language domains, translation requires some semantic understanding. For example, machine translation systems have been successfully used in Canada to translate weather forecasts, a domain with a limited scope, from English to French. A similar attempt to develop a machine translation system for aviation hydraulics was abandoned after three years. (Klein 1999).

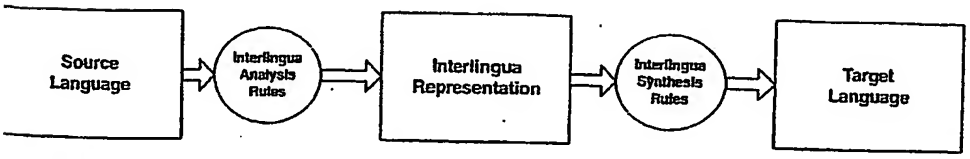


Figure 8.6 Semantic representation in a common meaning representation scheme distinguishes the interlingua approach from the direct and transfer methods.

A third problem with machine translations is known as lexical gaps. This occurs when one language has a single word that can be translated into two or more words in a target language. For example, the English word *know* translates into both *savoir* and *connaitre* in French and into *wissen* and *kennen* in German. Determining the correct translation requires a semantic understanding not usually found in current machine translation systems. One approach to dealing with this problem is to allow a user to choose among alternate translations.

Document Storage Options

Translated documents add another dimension of complexity to the document warehouse. We now have multiple versions of the same document in the sense that each translation conveys the same semantic content. Some of the options that warehouse designers have are:

- Storing the original text and all translations
- Storing the full text of the original and only a summary of the translated document
- Storing only a summary of all versions, including the original
- Storing the translation but not the original

How you choose between these and other options depends upon the importance of the document, the cost of retranslating if necessary, and any potential quality control and legal issues.

- The simplest solution—from a design perspective—is to store the full text of the original document as well as all translations. The advantage of this choice is having the original on hand in case there are questions about the translation at a later time. It also provides translations of the full text, so readers would not need to translate the entire document if only a translated summary was stored.
- Storing the full text of the original and translations of the document summary is another option. In this scenario, space use is optimized, and readers can still get the gist of a document from the summary. Since only summaries are translated, this could reduce the translation load by as much as 80 percent. Similarly, the original document may not be important enough to warrant storing the full content in the warehouse, and in this case, a summary-only scenario is appropriate.
- Finally, having news stories, press releases, and other noncritical documents in the original language may not add any substantive value to a doc-

ument warehouse. A mobile phone manufacturer announcing a new line of products in Finnish may not need to be included in the warehouse when an English version meets the needs of end users.

- In general, the most important documents—such as contracts, legal opinions, and government regulations and reports—warrant multiple language versions in a document warehouse, especially when machine translation techniques are used. Readers may get the main point of a document from a machine-translated rendition, but details and subtle points can easily be missed or misrepresented in an automatically generated translation. Because of these limitations, it is important for users to understand how the document was translated and any shortcomings of that method.

Indexing Text

Indexing text allows us to efficiently search for documents relevant to a query without examining entire documents. In this way, text indexing is similar to conventional database indexes, which allow us to forgo full table scans for more efficient retrieval of rows of data. The two types of indexing we are primarily concerned with are full text indexing and thematic indexing.

Full Text Indexing

Full text indexing occurs automatically in many text analysis tools when documents are loaded. Indexes will generally record information about the location of terms within the text so that proximity operators, as well as Boolean operators, can be used in full text queries. The most common operators in text queries are:

- Boolean Operators
  - AND
  - OR
  - NOT
- Proximity Operators
  - NEAR
  - WITHIN

Indexing supports Boolean operators by allowing those operations to be performed on the indexes without full document searching. Since the indexes

maintain information about the position of words within a text, proximity operators can also make use of indexes. The NEAR operator can be implemented in several ways. First, it can return a score relative to the distance between two specified terms. It can also be limited to searching for pairs of words with a maximum number of words between them. Finally, the order of terms in a NEAR query may or may not be relevant.

To maintain the efficient implementation and use of full text indexes, frequently used words, called stop words, are often ignored. Stop words appear so commonly in discourse that they do not add any value to document searching and can be safely ignored.

Some tools, such as Megaputer's Text Analyst automatically calculate the lexical affinities that measure the co-occurrence of words. Words that appear together such as *real estate*, *mortgage rate*, and *data warehousing* will be identified as lexical affinities. Using lexical affinity measures can improve full text searching by helping to disambiguate words with multiple meanings such as *bed* in *flower bed* or *queen-sized bed*.

hematic Indexing

Thematic indexing depends upon the use of thesauri. A thesaurus is a set of terms that define a vocabulary and are organized using relationships. It provides a hierarchical structure that allows text mining tools to quickly find generalizations as well as specializations of specific terms. The ISO-2788 standard for monolingual thesauri is the most commonly used standard for thematic indexing and consists of four main components:

- Thesaurus
- Indexing term
- Preferred term
- Nonpreferred term

Indexing terms are either a single word or a compound term representing a concept in the thesaurus. Preferred terms are the terms used when indexing concepts. For example, *automobile* could be the preferred term for *car*, *van*, and *minivan*, which are considered nonpreferred terms.

Preferred terms are organized hierarchically. Nonpreferred terms are tied to the hierarchical structure by their reference to a preferred term. Preferred terms are related to each other by relations that define the hierarchy. The ISO-2788 standard defines the following relations:

- USE: The term that follows is a preferred term.
- UF: The term that follows is used for a preferred term.
- Top term: This specifies the name of the broadest class to which a term belongs.
- BT: This defines a broader, generalized term for a specified word.
- NT: This defines a narrower term that specifies another term.
- RT: The related term operation associates words that are not synonyms or quasisynonyms of a given term.

Some tools, such as Oracle interMedia Text, are preconfigured with a thesaurus and can be used immediately to thematically index text. Others tools—and some text mining applications—will require custom thesauri. Figure 8.7 shows a sample thesaurus using standard terms and relations.

With a thesaurus, applications will be able to search by topic as well as by full text. It is highly recommended that all document warehouses provide this basic service. Thematic indexing reduces the poor precision and poor recall associated with polysemy (words having multiple meanings) and synonymy (multiple words for the same meaning).

```
Company
NT Corporation
NT Sole Proprietorship
NT Limited Liability Partnership
SYN For-profit Organization
Organization
NT Company
NT Non-profit Organization
NT Government
Government
NT Federal Government
NT Regional Government
NT Municipal Government
Regional Government
SYN State Government
SYN Provincial Government
```

Figure 8.7 A sample thesaurus in ISO-2788 format.

Full text and theme indexing are usually implemented to support ad hoc searching, but they also provide the basis for document classification. By looking at the pattern of words and themes, we can develop a rough partitioning of documents into a predefined set of groups. Examples of such groups include:

- Industry sector news stories
- Regulatory notices
- Project-related documents
- Product-specific technical documentation and manuals
- Financial reports

These rough partitions can be further refined as necessary. For most document warehouses, two types of classifications will be used: labeling and multidimensional taxonomies.

Labeling

Labeling is the process of assigning a dominant theme or topic descriptor to a document. The labels chosen may be domain dependent or in general categories— such as the ones found in Oracle interMedia Text's knowledge catalog. For finer classifications, multiple labels can be assigned along with weights. For each document, a list of labels and weights are assigned:

```
Document = [   (label1, weight1),
               (label2, weight2),
               . . .
               (labeln, weigthn)]
```

The labels and weights can be used with text querying tools to specify minimum thresholds when looking for documents. For example, the following code can be used to return the document identifier and title of documents about currency exchanges with at least a weighting of 0.5:

```
SELECT
  Document_Id, Title
FROM
  Documents
WHERE
  ABOUT(text_column, 'currency exchange') >0.5
```

Labels without weights can also be used to populate the SUBJECT field of the Dublin Core metadata set kept for document. Ideally, the document creator will

specify subject labels, but if these lables do not exist or do not conform to the preferred terms used in the document warehouse thesauri, then classification labels can be assigned.

Labeling—How It Works

Successful labeling depends upon three types of data:

- Word frequency statistics
- Morphological knowledge
- Type-specific terms

Two word frequency statistics are necessary: relative frequency and absolute frequency. Relative frequency measures the number of times a word appears in a document. Absolute frequency measures the number of times a term appears in a set of documents. Depending on the classification tool, absolute frequency might be calculated over a broad range of documents and the statistics provided along with the tool. In other cases, tools can be trained by using sample documents provided by document warehouse designers and text miners.

Morphological knowledge is used to determine the preferred (or canonical) representation of a term. For example, the canonical form of *eat*, *ate*, *eats*, and *eating* is *eat*. Morphology is used to eliminate the variations that occur in language such as tense, plurality, and, in some languages, noun declinations and verb conjugations. So no matter how the root of the word is modified to meet the grammatical rules of the source language, it will be identified as a single term.

Type-specific terms are used to augment general lexicons and thesauri. These extra terms include names of cities, states, provinces, and other geographic references as well as common abbreviations, names of clients and customers, and other company- or domain-specific terms.

Once morphological analysis renders words into a standard canonical form, relative frequencies can be calculated. The most common measure for determining the weight of a term in a document is the inverse document frequency measure. The basic idea behind the measure is that high weights should be assigned to terms that appear in few documents because these are good discriminators. Since relative frequency measures the number of times a word appears in a document, its weight will be proportional to the relative frequency. Terms that appear in many documents have a high absolute frequency and indicate poor discriminators. In these cases, the weight is inversely proportional to the absolute frequency.

The combination of terms and weights has proven to be a powerful technique for classifying documents. One limitation of labeling, though, is that it does not generalize. For example, a document labeled *automobiles, trucks or buses* or *rail*

transportation is also about *ground transportation*. Without the use of a [illegible] one cannot easily query for generalized concepts such as ground transportation.

Multidimensional Taxonomies

The idea of a multidimensional classification structure is well known to data warehouse practitioners. Ralph Kimball and others have developed the multidimensional model into an effective tool for organizing large quantities of numeric data in data warehouses. Multidimensional models allow us to quickly and easily target a subset of the database that interests us, using major structural categories, such as time period, customer, product, and location. Similarly, with multidimensional taxonomies, we can quickly and easily target a subset of a document set by using classification categories, as shown in Figure 8.8.

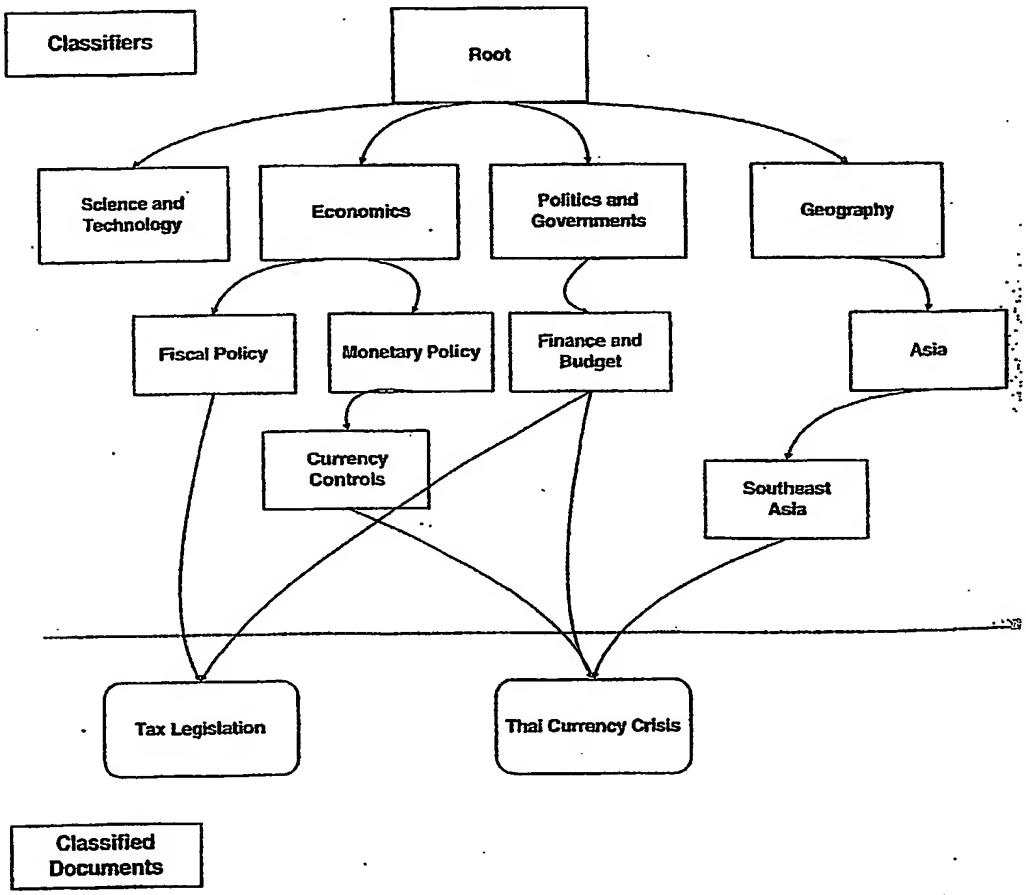


Figure 8.8 A partial sample taxonomy for classifying a broad range of documents.

Taxonomies can be created using specialized taxonomy-generation tools or with a combination of clustering and feature extraction, as described in Muller et al. (1999). Given a taxonomy, we can classify documents with both specific terms, as in the case of labeling, and hierarchical categories. The net effect is equivalent to drilling-up hierarchies in a multidimensional data warehouse. Thus, we can query for documents about *ground transportation*, and we can find documents about *automobiles* and *rail transportation*.

When dealing with taxonomies, it is useful to distinguish two concepts: the intention of a term and the extension of the term. The intention of a term describes the term abstractly, by relating it to other abstract terms. For example, *automobile* is a type of *ground transportation*. The extension of a term is the set of documents (in our case) that are about that particular term. For example, the extension of the term *automobile* might be documents with document IDs 1001, 2387, 11183, and 93321. The extension, thus, points to actual documents which instantiate the concept of *automobile*. With these definitions in hand, we can now proceed to discuss how multidimensional taxonomies are used within the framework of the document warehouse.

From the classification problem perspective, multidimensional hierarchies classify particular documents (the extension) into multiple categories at multiple levels of generality (the intention)—thus providing a richer classification scheme than labeling alone.

Document Clustering

Document clustering may be useful for some applications, such as quickly finding similar documents and exploring the macrostructures of a large collection of documents. Clustering can also help identify duplicate documents in the warehouse so they may be removed. Unlike classifications, clustering does not presume a preexisting set of terms or a taxonomy that is used to group documents. Instead, groups are created on the basis of the features of documents within the set of documents being clustered. Although this technique is not as common as thematic indexing or summarization, it may prove useful to some.

Many techniques have been developed for document clustering, but we will concentrate on three main types:

- Binary relational clustering
- Hierarchical clustering
- Self-organizing maps (SOM)

Binary relational clustering partitions a set of documents into groups, with each document in a separate group. Hierarchical clustering groups documents at multiple

levels, providing drill-up and drill-down navigation. Self-organizing maps are especially useful for document sets covering a broad range of topics, such as e-mails.

Binary Relational Clustering

Like other clustering techniques, the main objective of binary clustering is to group documents so that the similarity measures between documents in a cluster is maximized, while the similarity between documents in different clusters is minimized. The dominant features of binary relational clustering are:

- Clusters are flat.
- Documents are in only one cluster.
- Clusters correspond to a single topic.

Binary relational clustering works by assigning documents to a single cluster, much as labeling assigns one classification to a document. Like labeling, each cluster corresponds to one topic, which is basically the set of common features shared by all documents in a cluster. For example, a cluster with documents about Windows NT, Windows 95, DOS, VMS, UNIX, and Linux corresponds to an operating system cluster. As Figure 8.9 shows, binary relational clustering groups documents on the basis of similarity threshold and a predefined number of clusters, and does not guarantee a balanced distribution of documents over all clusters.

Hierarchical Clustering

Hierarchical clustering groups documents together according to similarity measures in a tree structure. As Figure 8.10 shows, documents can be in multiple clusters in a hierarchical clustering scheme. Rather than finding the single best match between a document and a cluster, hierarchical clustering algorithms iteratively group documents into larger clusters.

The basic algorithm works as follows: First, assign each document to its own cluster. These are the leaves of the tree. Then create the second level of the tree by merging two clusters at a time, grouping them according to similarity. Create the third level by grouping pairs of clusters from the second level, and so on, until all groups have been merged into a single cluster at the root of the hierarchy.

One of the advantages of hierarchical clustering is that it supports browsing by drill-down and drill-up operations.

Self-Organizing Map Clustering

A third clustering technique uses a neural network to map documents in document sets that have many possible topics (that is, the document space is highly

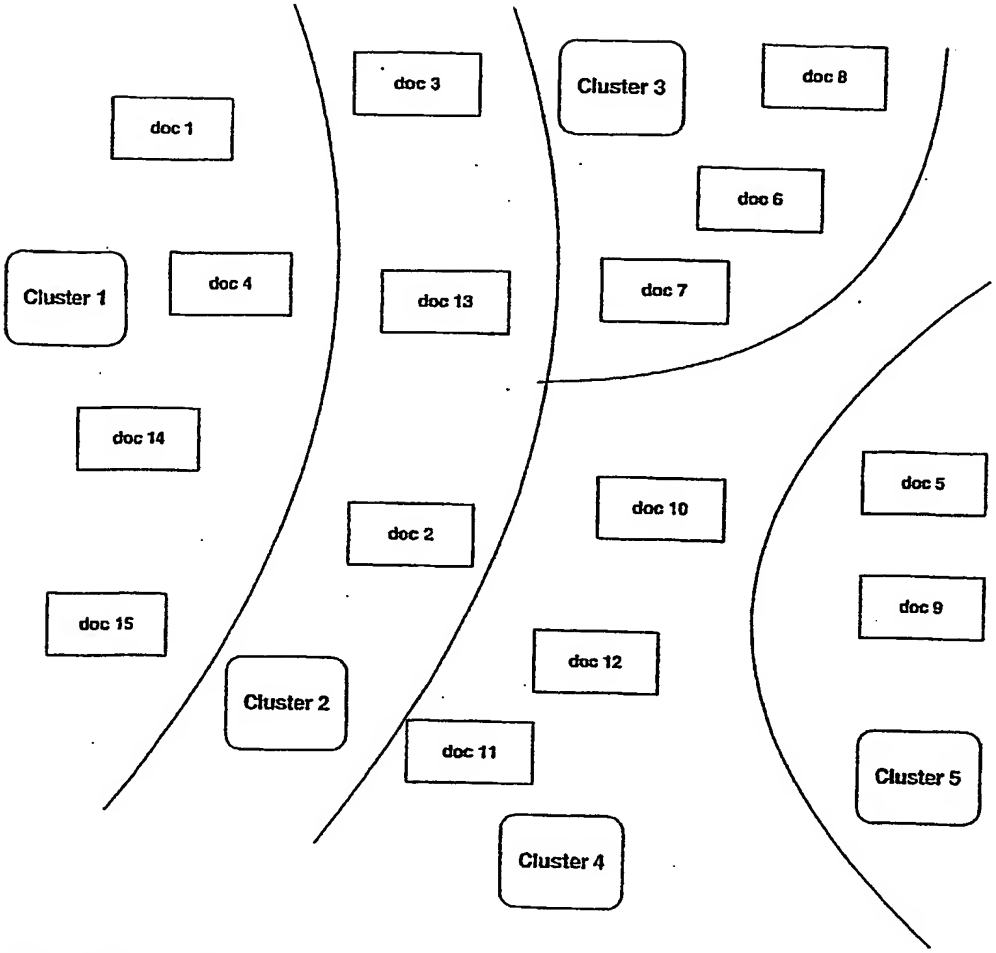


Figure 8.9 Binary relational clustering renders a flat partitioning of a set of documents.

dimensional), where each document has only a small number of those topics (that is, the document space is sparse).

Like the other clustering technique, self-organizing maps (SOMs) depend upon a similarity measure. Unlike the other techniques that compare documents to each other, SOMs compare the similarity of a document to a point on a two-dimensional grid, as depicted in Figure 8.11. The grid is created initially and populated with weighted feature vectors. The similarity measure compares the distance between a document and each the feature vector, corresponding to the point on the grid. After finding the closest match, the algorithm adjusts the weights of the feature vectors at the grid point to move it a little closer to

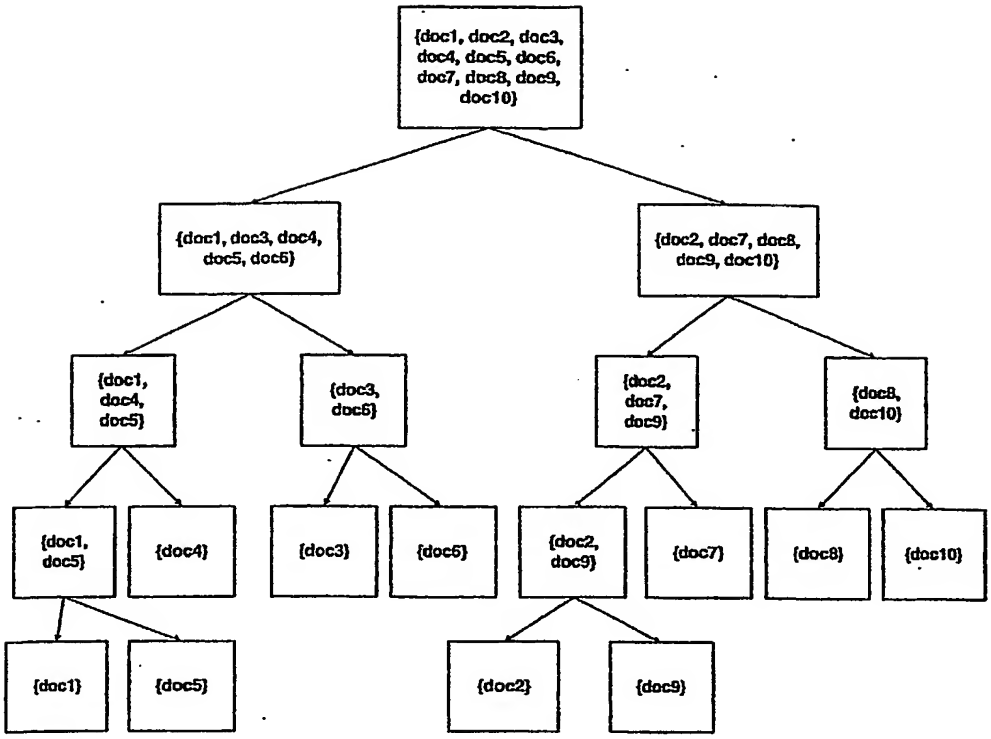


Figure 8.10 Hierarchical clustering groups documents into multiple clusters.

the document just added to the cluster. The amount of adjustment is controlled by a rate of learning parameter specified when the clustering program is run.

SOMs have been used successfully with large newsgroup collections, which are generally considered difficult to analyze because they are frequently filled with short, ungrammatical pieces of text (Kohonen 1998).

Clustering techniques will prove useful when trying to understand the overall structure of a document set and for some maintenance operations, such as detecting duplicates.

Summarizing Text

The goal of summarization is to reduce the length and complexity of a document while maintaining its meaning. The two basic methods of summarization are summarization by abstraction and summarization by extraction. When we humans summarize, we generally read the entire text, develop an understanding of the main ideas, and then write a coherent summary of the text. This is

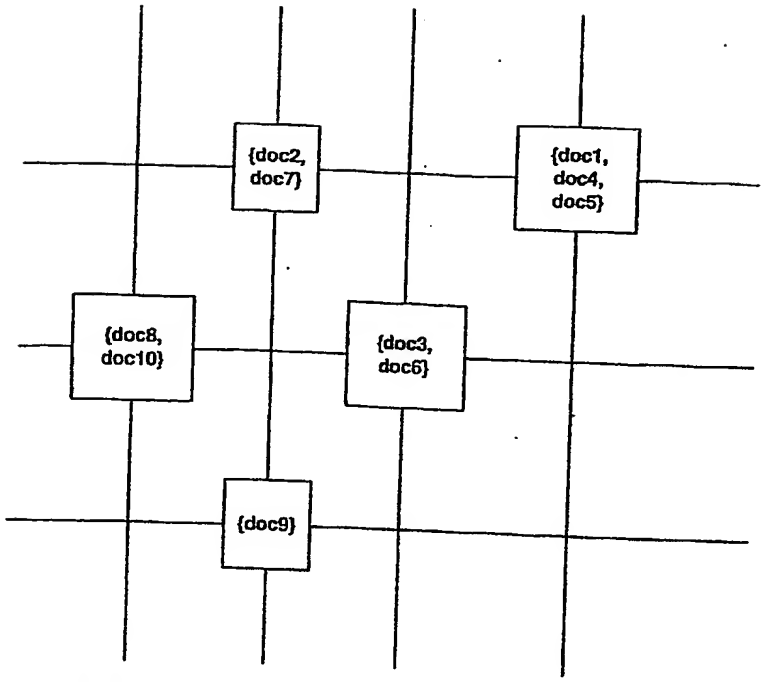


Figure 8.11 Self-organizing maps use two-dimensional grids to organize the clustering of documents.

summarization by abstraction and is beyond the abilities of automated methods. Summary by extraction works by taking key parts of the text and building a summary without understanding the meaning of the text. Three distinct approaches to summary by extraction have been proposed:

- Paragraph extraction
- Sentence extraction
- Sentence segment extraction

Each technique has distinct advantages, as we shall now discuss.

asic Summarization Methods

All three methods determine the most important terms using the same techniques used for document classification and clustering. In the case of paragraph extraction, entire paragraphs are ranked according to the presence of important terms in them, and the most important paragraphs compose the summary. The primary advantage of this method is that the summaries are the most rhetorically coherent of the three approaches.

Sentence extraction works similarly, but at the sentence level instead of the paragraph level. In this approach, less text is retained in the summary since unimportant sentences within a paragraph are discarded. Sentences are usually ordered according to their relative weights, so it is not uncommon to lose rhetorical consistency. For example, a sentence that begins with *In conclusion ...* could appear before a sentence that begins, *First, there is the issue of...*' This problem can be avoided by ordering sentences in the same order in which they appear in the document, rather than by their weights.

Sentence segmentation drills down even farther to work at the clause level. With this technique, a sentence is divided into segments by looking for cue phrases. Each segment conveys a single idea, such as *interest rates are rising.* Segments are separated by cue phrases like *because* or *that*, as in *interests rates are rising because the Federal Reserve is concerned about inflation.* The primary advantage of sentence segmentation is that is removes clauses within sentences that do not convey important information. Like sentence extraction, this technique can suffer from poor rhetorical cohesion.

Dealing with Large Documents

While all three methods for summarization by extraction will produce suitable summaries for most texts, there are special issues that must be addressed with large documents. Prior to summarizing, large documents may need to be pre-processed in one of several ways, through:

- Document partitioning
- Tabular data extraction
- Targeting structured elements

Document partitioning uses knowledge of document types to identify semantically distinct sections of document. Many business and government documents contain both text and numeric data that is essential to understanding the meaning of text and needs to be addressed when summarizing. Finally, semistructured texts can provide additional clues about important elements of a document and may need to be extracted during document parsing.

Document Partitioning

Large documents are usually divided into logical sections. For example, a business plan will provide a discussion of the business organization, financial plans, and marketing information. Project plans might describe the business problem solved by the project, the proposed team structure, duration and funding. Summarizing across logical boundaries can cause problems.

when the relative importance of a document section is not reflected by the section's length.

Recall that the importance of a term is measured by its relative frequency in a document and measured inversely proportional to the term's appearance in the larger document set; thus, the number of times a term appears in a document will control its measure of importance. For example, the marketing section of a business plan may be a dominant section of a business plan so terms such as *market segment* appear frequently. Since this is a relatively uncommon term across all documents, it will have a low absolute frequency and thus be considered an important term in the document. Now a term such as *long-term debt* may appear just as infrequently over the set of all documents and, thus, have a similar absolute frequency to the term *market segment*. However, if the financial plan section comprises primarily short texts and tabular numeric data, then terms such as *long-term debt* will have a low relative frequency, making their appearance in the summary less likely.

To avoid this problem, semantically independent document sections may need to be separated into separate documents before summarizing. The result of the individual summarization operations can then be merged to create a semantically adequate document summary. XML is an ideal tool for document partitioning.

Tabular Data Extraction

Since numeric data can often explain a fact much more concisely than a verbal description, tables of data are often embedded within documents to help convey a point. Summarization techniques are not designed to deal with numeric data, so it is best to extract these tables before summarizing and merge the extracted tables with the summarizer's output.

Table extraction can be done either with generic programs for report reformatting (sometimes called "screen scrapers") or with custom programs. If the tabular data is relatively well structured and consistent across a large number of documents, then the screen-scraping approach may be the most efficient. If tabular data will vary in location and complexity, then a scripting language with strong support for regular expressions, such as Perl or Python, can provide the flexibility needed to build a robust extraction routine, but at the expense of writing and maintaining a custom program.

Targeting Structured Elements

With the increasing popularity of XML and derived standards, we can expect to find more and more documents in the warehouse with these structuring

elements. Document warehouse designers can take ... ments in several ways.

First, rather than using a summarization by extraction program to create a summary, one could extract salient entities from the XML document, such as an executive summary and the conclusion. Multiple summaries could also be created targeted to different audiences. Financial analysts could be provided with an executive summary and key financial indicators, while marketing executives might prefer the executive summary and an automatically generated summary of the other major sections.

Second, only particular sections of the document might be summarized. This could lead to some improvement in the quality of the summary but it could also significantly reduce the amount of text that must be analyzed by the summarizer.

Third, summaries can replace the contents of some sections of the XML document and thus keep the benefits of the semistructured document, while reducing the size and complexity of the document. This approach is useful when some entities are significantly longer than their relative importance warrants.

Conclusions

Loading and transforming documents is a critical and complex operation in document warehouses. The initial steps begin with preparing for documents in multiple languages and in multiple character sets. Preprocessing steps might include converting character sets, translating documents, extracting tabular data and identifying key entities in semistructured texts.

Main processing steps include indexing documents, both by full text and by theme or topic. Document metadata can then be augmented by adding topic labels to support metadata-oriented searching. Since documents cover multiple topics, the use of multidimensional taxonomies can greatly improve browsing by end users.

Summaries are an important aid to end users because they reduce the complexity of texts without losing significant amounts of information. Automatic summarization is not without its limits, but preprocessing large documents and documents with significant amounts of text can improve the quality of the final summary.

Classification and clustering do not strictly transform documents but augment what is known about documents by making implicit features, explicit. The end result is that users have explicit representations of implicit relationships between documents.

A number of operations are required during the load and transformation stage—managing these operations, and others, is the topic of the next chapter.